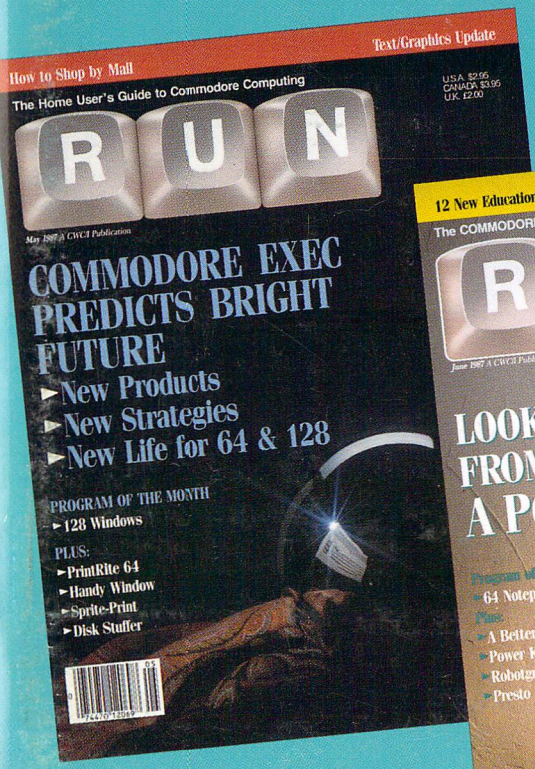
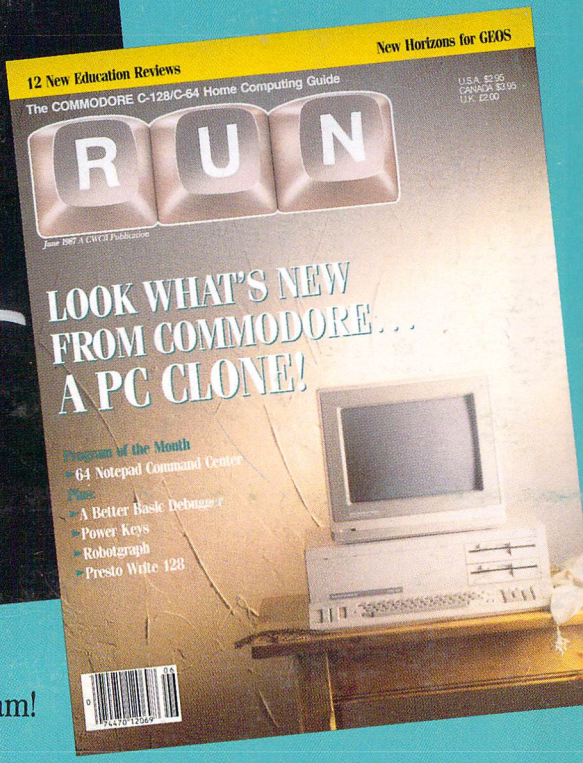
May/June 1987 Edition

RE RUN

RUN Programs on Disk





Inside:
Super Bonus Program!

Introduction

May-June '87 ReRUN

It can't be done. That's the theme and motivating factor behind the creation of this May-June ReRUN disk, but more on that subject later.

I and many people I know consider late spring and early summer to be the best time of the year. Although the warm afternoons tend to make the beaches irresistibly tempting, set aside some time in the evenings and rainy weekends for your Commodore and try out the exceptional programs on this ReRUN disk.

Foremost on our May-June disk is Presto Write 128, a superbly executed Basic 7.0 program written for the C-128. It effectively turns your C-128 and printer into an intelligent typewriter that allows for viewing and editing of each line of text before it gets sent to the printer.

Next on the list of C-128 programs, you'll find 80-Column Custom Windows—one of those programs you wish you owned when you see it in action on someone else's computer. It lets your C-128 create dazzling, Amiga-like windows that appear and vanish as quickly or slowly as you wish. Handy Window—for the 128 or 64—consists of eight routines that furnish you with most of the bells and whistles found on fancier upgrade desktop accessories.

If you're a C-64 owner in search of a programming utility that offers a trace function similar to Basic 7.0's TRON (TRace ON) and TROFF (TRace OFF) commands, look no further. Basic Bug Trap makes TRON, TROFF and several other debugging aids available on your C-64. It's one of those rare gems that all C-64 users, regardless of their programming expertise, want to add to their software libraries.

By now, nearly every C-128 owner has acquired a program that allows him to access the C-128's numeric keypad from C-64 mode. However, if you still suffer from the lack of a C-64 mode Keypad program, your days of searching and yearning have ended. May's Mega Magic—C-128 Keys in C-64 Mode— gives your C-64 mode those keys.

Returning to my initial statement, I've found that many Commodore computerists like to hear someone mention a particular programming objective and then say it can't be done. Accepting the challenge, the

hardcore computerist then spends every free moment trying to prove otherwise. In this example, it's printing those elusive sprites. Sure, it wasn't easy, but Sprite-Print is proof that those sprites that look so good on the screen can look good on paper, too.

Here's another "it can't be done"-ism. It has been generally accepted that only 144 files fit on one side of a 1541-formatted disk. Naturally, one of the more dedicated Commodore users sought to do something about that problem, and Disk Stuffer provides another answer. It consists of a three-program process that modifies your disk to accept up to 288 files.

Producing clear, legible characters with Commodore 803 or 1525 dot matrix printers isn't easy, because they lack the true descenders needed to produce readable letters. PrintRite 64 gives your printer those much-needed descenders. It also gives them to your RUN Script documents by working with version 1.0 of RUN Script 64.

The 64 Notepad, published in *RUN* in September 1986, with an update in January 1987, is one of those utilities that folks just seem to fall in love with the first time they use it. We knew it was an exceptional program, but the positive reader response caught us by surprise. We go one better in this ReRUN with 64 Notepad Command Center, the best Notepad enhancement to date by programming wizard Bob Kodadek. Included are two other programs—Notepad Boot, for booting up Notepad, and Notepad Save, for saving Notepad to disk as a binary file.

John Ryan, another programmer quickly earning legendary status in Commodore circles, provided us with Power Key, an easy-to-use utility that changes screen colors and toggles Quote mode in an instant.

Computer-generated music has just picked up a catchy cadence with Drummodore, a 64 program that converts your computer into—you guessed it—a drum machine.

This would hardly be a ReRUN disk without an educational program, and you'll find it in Robotgraph, which creates bar graphs to show relative values. A little robot makes the process entertaining by pulling up each bar.

Finally, there's this month's bonus program. Written by Daniel Miller, Puzzler's Choice contains not one, not two, but three mind-challenging games. Be careful of this one if you're stress-prone, for it's difficult.

I hope you enjoy the programs; don't hesitate to write or call if you have problems or questions about ReRUN.

Tim Walsh
Technical Editor
RUN *magazine*

How To Load

Loading from Menu

To get started, C-64 users should type LOAD "MENU 64",8 and press the return key. When you get the Ready prompt, the menu is loaded and you should type RUN to see a list of the programs on your disk. C-128 users need only press the shift and run-stop keys. When all the programs are displayed on the screen, you can run the one you select by pressing a single key.

Loading from Keyboard

If you do not wish to use the menu program, follow these instructions.

C-64:

To load a C-64 program written in Basic, type:

LOAD "DISK FILENAME",8

and then press the return key. The drive will whir while the screen prints LOADING and then READY, with a flashing cursor beneath. Type RUN and press the return key. The program will then start running.

To load a C-64 program written in machine language (ML), type:

LOAD "DISK FILENAME",8,1

C-128:

All C-64 programs can be run on the C-128 as long as your computer is in C-64 mode.

All C-128 programs are clearly labeled on the directory page. Your C-128 *must* be in C-128 mode to run these programs.

To load a C-128-mode program, press the F2 key, type the disk filename and then press the return key. When the program has loaded, type RUN.

Making Copies of ReRUN Disks

Some of the programs on your ReRUN disk have routines that require you to have a separate disk onto which the program writes or saves subfiles. In order for you to use these programs, you will first have to make a copy of the original program onto another disk that has enough free space on it to hold these newly written subfiles.

If the program is written in Basic, it is simple to make a copy of the program. Just load the program into your computer following the procedures outlined above, and then save the program back onto a separate disk that has plenty of free space for extra files.

If the program is written in ML, copying is not so simple. You cannot simply load and save an ML program. In this case, you'll need to use a disk-backup utility program, such as the one on your Commodore Test Demo disk.

Directory

Page	Article	Disk Filename	File Type
	*	MENU 128	BASIC
		MENU 64	BASIC
1	* Presto Write 128	PRESTO WRITE 128	BASIC
3	* 80-Column Custom Windows	WINDOW 128	BASIC
		WINDOW DEMO	BASIC
		CUSTOM WINDOWS	ML
7	* Handy Window	HANDY WINDOW 128	BASIC
		HANDY WINDOW 64	BASIC
11	Basic Bug Trap	DEBUGGER 64	BASIC
		128 KEYS IN 64	BASIC
15	Sprite-Print	SPRITE-PRINT 1	BASIC
		SPRITE-PRINT 2	BASIC
		SPRITE-PRINT 3	BASIC
		PDUMP/1525	ML
18	PrintRite 64	PRINTRITE 64	BASIC
21	Disk Stuffer	LONG DIRECTORY1	BASIC
		LONG DIRECTORY2	BASIC
		LONG DIRECTORY3	BASIC
23	64 Notepad Command Center	NOTEPAD DOS	BASIC
		NOTEPAD BOOT	BASIC
		NOTEPAD SAVE	BASIC
26	Power Key	POWER KEY	BASIC
29	Drummodore	DRUMMODORE	BASIC
32	Robotgraph	ROBOTGRAPH	BASIC
34	Mega-Magic, May	128 KEYS IN 64	BASIC
35	Mega-Magic, June	HI-TECH SPRITES	BASIC
36	£ Puzzler's Choice	PUZZLER'S CHOICE	BASIC

* C-128 program
£ Bonus program
NOTE: Read articles before running these programs!

ReRUN Staff

Technical Editor: *Tim Walsh*
Managing Editor/Production: *Swain Pratt*
Copy Editor: *Peg LePage*
Proofreader: *Harold R. Bjornsen*
Design and Layout: *Karla M. Whitney*
Typesetting: *Doreen Means, Beth Krommes, Ken Sutcliffe*
Special Products Director: *Jeff DeTray*
Special Products Manager: *Vivian Mattila*
Special Products Assistants: *Debbie Bourgault, Robyn Johnson*
Direct Marketing Coordinator: *Debbie Walsh*

Presto Write 128

By Bob Kodadek

RUN It Right

C-128; printer

Presto Write 128 is a mini word processor you can use for typing in, editing and printing out small documents such as notes, short letters, envelopes and labels. It's better for this purpose than a typewriter, because it enables you to edit, and it also avoids the complicated, time-consuming command structure of a full-fledged word processor.

By taking full advantage of the windowing, editing and reverse video capabilities of the 8563 display chip, Presto Write displays on the screen exactly what will be printed on paper. The command line interface (CLI) lets you easily set and alter the margins at any point in your document and number lines within the document for quick position reference.

I wrote part of the program in Basic 7.0 to demonstrate a variety of its commands. However, even this updated version of Commodore Basic is slow for word processing, because it can't build or print long strings of text quickly. So, I also wrote part of the code in machine language. I've found that adding machine language to Basic 7.0 can produce marked increases in speed.

USING PRESTO WRITE

After you load Presto Write, enter your own data in line 20, which defines the F1 key to automatically print your name, address, city, state and zip code at the current left margin. Then save the program to a work disk.

The CLI uses the back-arrow character as a command identifier. Any time this character appears in your text, a legal three-character command must follow or an error will be generated.

I've written the following three commands into the program:

←SET Set left and right margins.
←CLR Clear screen and start a new page.
←END Exit the program.

You can add more commands in lines 70–110 of the program if you wish.

After you type a ←Set command into your document and press the return key, Presto Write will prompt you for margin settings. To choose the default settings of 5 left and 75 right, press return again in answer to each of these prompts. These settings will produce a one-inch margin on either side of your document.

Presto Write's screen editor creates lines of up to 80 characters. To manipulate text, use the cursor, delete and insert keys, plus escape-key sequences such as escape-D, which deletes an entire line. See your *C-128 System Guide* or *C-128 Programmer's Reference Guide* for descriptions of the escape functions available.

You can use the caps-lock and home keys with Presto Write, but never press the home key twice in succession or you'll exit the editing window.

After you enter a line of text and check it for errors, press return to print it out. If you forget to turn on your printer, the program will alert you and let you correct the situation without having to type the line over again.

PASSING PARAMETERS

Basic 7.0 really shines in passing parameters (sharing data) between Basic and machine language. With the SYS command, you can place values directly into the accumulator, x, y and status registers before the machine language routine is called from Basic and without using cumbersome Peeks and Pokes. In addition, Basic 7.0 has an undocumented RREG command that reads the 8502 registers and places their values into Basic variables. The format for RREG is RREG A,X,Y,SR.

For example, in line 70, the SYS 2816,L command loads the accumulator with the value of the current left margin setting. Then the machine language routine stores this information to be used later in formatting the printed output. The machine language routine returns to Basic with one of three values held in the accumulator register.

The RREG A command then places this value into the Basic variable A. If A is found to equal 95, a CLI command character was detected during input, and the output routine aborts; or, if A equals 255, the printer isn't present. Otherwise, a zero is returned in the accumulator, indicating that no error occurred and no corrective action is necessary. ■

80-Column Custom Windows

By Frederick Goddard

RUN It Right

C-128

I've long envied the Amiga's ability to create temporary windows that allow you to view the disk directory or list a program without disturbing the original screen display. To take advantage of the C-128's similar capabilities, I designed 80-Column Custom Windows, a utility that gives C-128 owners access to Amiga-like windowing feats. It will not turn your C-128 into an Amiga, but it will allow you to create temporary windows with either Basic or direct keyboard commands.

This utility is lightning fast and provides up to eight bordered windows with a shadow, or 3-D, effect. You can enter commands in the windows, list and change lines in a program, display a disk directory and enter the machine language monitor. 80-Column Custom Windows doesn't alter your computer's operating system in any way. When you're finished with the windows, you can erase them, leaving the original screen intact.

GETTING STARTED

Window 128 creates a machine language program called Custom Windows on the disk. When you load and run Custom Windows, the screen clears and a title box appears, telling you that the program is active and you have 96,765 bytes free. This machine language version loads and runs like a Basic program, so if you save it as an autoboot program, be sure to identify it as being in Basic.

Window Demo is a short program that shows you how you can use Custom Windows. Once you run the Custom Windows program, run Window Demo to see an amazing demonstration.

HOW IT WORKS

Custom Windows occupies specific memory locations at 7169–8785 ($1C01–$2251). The beginning of Basic is moved up to 16385 ($4001) with a Graphic1:Graphic5 command. Some of the variable memory in Bank 1 is reserved to store the screen text and attributes underneath the windows you create. The default value for this allocation is 16K, but you can poke a different amount if you wish (as little as 4K or as much as 32K). It takes 2K of memory to create a temporary window that covers half the screen, so the default value of 16K is ample.

The program thus lets you open up to eight 80-column temporary windows at a time; when the windows are closed, the original screen will reappear intact. Each window has a number (1–8) displayed in the upper-left corner. The procedure for opening a window is the same whether you use Basic or direct keyboard commands.

To open a window, you first set the color. In Basic you can use either the Color or Print command to do this; in Immediate mode, you use the control and logo keys, along with number keys 1–8. You get the most attractive results if you choose light shades. (Dark shades nullify the shadow effect.)

Next, set the size and location of the window. Use the Window command in Basic or position the cursor at the top-left corner of the new window and press ESC T; then position the cursor at the bottom-right corner and press ESC B. Be sure to allow two extra columns and rows for the border.

Third, open the window, using the commands listed below. When you issue a command to close a window, the last window opened will be closed, and the screen will display the text that existed before that window was opened.

To use Custom Windows in a Basic program, insert the following lines at the beginning of the program:

```
2 GRAPHIC1:GRAPHIC5
3 BLOAD"Custom Windows"B0,P7169
4 X=16:POKE 7194,X:SYS7197:CLR
```

Line 2 sets the start of the Basic program at $4001 to allow space for Custom Windows. Line 3 loads Custom Windows. Line 4 sets the number of kilobytes to be allocated for window memory. (The default is 16; change the value of X to another number from 4 to 32 to allocate less or more Bank 1 memory for this purpose.) Line 4 also executes SYS 7197 to activate Custom Windows and set variable pointers.

COMMAND OPTIONS

Activate the program. Clears the screen, prints a title box indicating the number of bytes free, sets the pointers to the next window, redefines the run key, and sets an interrupt wedge for direct keyboard commands. In Basic, use SYS 7197. You can activate the program without printing the title box with SYS 8442. If you're going to use the program exclusively as a Basic program subroutine and don't want to activate the interrupt wedge for direct keyboard entry, use SYS 8220.

Open a window. In Basic, use SYS 7200. Direct from the keyboard, simultaneously press the control and stop keys.

Close a window. In Basic, use SYS 7203. Direct from the keyboard, simultaneously press the logo and stop keys. (You should always execute a Print command after you close window 1 from Basic.)

Reset pointers and enable interrupt. In Basic, use SYS 8364,1. Direct from the keyboard, simultaneously press the shift and stop keys. Resetting the pointers locks in existing windows and also sets the next window to be opened at 1.

Reset pointers and disable interrupt. In Basic, use the SYS 8364,0 command. Direct from the keyboard, simultaneously press the ALT and stop keys.

Reset (interrupt unchanged). In Basic, use SYS 8220.

Enable interrupt (pointers unchanged). In Basic, use SYS 8319.

Disable interrupt (pointers unchanged). In Basic, use the SYS 8371 command.

Open a directory window. Puts a special window on the screen designed for directory displays. The left edge of the window is in the column in which the cursor was located when the command was issued. You must issue a separate Directory command to display the disk directory in this window. In Basic, use SYS 8666. Direct from the keyboard, simultaneously press the control and ALT keys.

LIMITATIONS

If you try to open a ninth window, the program will ignore the command. It will also ignore a command to open a window if there isn't enough memory in Bank 1 to hold all the text and attributes under the new window, or if the window has fewer than four rows or eight columns.

When you issue a Close Window command, the last window opened will be closed. That is, if you have three windows on the screen, it is not possible to close window 2 before closing window 3. If you use Custom Windows in

a Basic program, you must, after closing window 1, issue a Print command to set the screen pointers before you can open any new windows.

When you use direct keyboard commands (as opposed to Basic commands), the program works through an interrupt wedge. This produces ghost cursors on the screen if you use different colors for the windows. These cursors won't affect the validity of the screen text, but they will be obvious.

You can minimize the ghost cursors by applying the following two rules. First, never open a window with the cursor in the left-most or right-most column of the window. A simple cursor-right key entry before the Control/Stop command generally will fix this. Second, always change the character color to the previous window's color and clear the current window before issuing the Close Window command (logo/stop).

Finally, you can load and run a program with Custom Windows activated as long as the program doesn't overwrite Custom Windows in memory locations 7169–8785. This means that you can't run programs that use 40-column, high-resolution graphics while Custom Windows is active. Before running any commercial software or programs that use such graphics, reset the computer by pressing the reset button or the stop and restore keys.

Eighty-Column Custom Windows is a very useful and attractive utility. I've put it to work as an autoboot program on all my disks, as it's very helpful when I'm writing programs or cleaning up disks. I hope you have as much fun with it as I have. ■

Handy Window

By Ian Adam

RUN It Right

C-64 or C-128

Handy Window is a collection of eight computer routines that give you full control of a memo pad, an accurate digital clock and an alarm, as well as access to the printer. Both the C-64 and C-128 versions are so easy and convenient to use that you'll wonder how you ever computed without it.

After you've loaded the version of the program appropriate to your C-64 or C-128 (40-column screen only), you can proceed with programming, or you can run a Basic program. You only have to remember that Handy Window uses just two keys: shift and control. Pressing them at the same time displays the Handy Window menu; the same combination removes the menu from the screen and returns you to the program you were in before.

GETTING STARTED

When you run Handy Window, it prompts you to insert a disk for the code. Insert any formatted disk, then press any key to create a file called .HANDY64 or .HANDY128 on the disk. This process takes about 30 seconds. You can place a copy of Handy Window on any of your other disks by running the program once for each disk.

The resulting file isn't a Basic program that you can load and run; it's a machine language file that must be executed. On the C-64, you execute the file with the following two-line program:

```
10 IF PEEK(49152)<>120 THEN
   LOAD".HANDY64",8,1
20 SYS 49152:NEW
```

On the C-128 in 128 mode, a different two-line program is required:

```
10 BLOAD".HANDY128"
20 SYS 4864
```

HOW TO USE IT

The Handy Window menu gives you eight choices, from which you make selections by pressing the corresponding keys. These are the eight options:

Memo Pad (A): Choosing A pro-

duces a blank screen on which you can type memos to yourself. All screen editing functions are active, including scrolling and screen clear, so be careful not to erase your message. Press the shifted control key to return to the mode you were in when you pressed A. If you choose A again, your notes will still be there, waiting for you.

Show Clock (B): The digital clock is always displayed when the menu is on-screen. If you choose this option, however, the clock will also be displayed when you leave the menu to enter the Memo mode or return to your original program. A bull's-eye appears beside the letter B to acknowledge your selection.

Hide Clock (C): When you select this option, the only immediate result is that the bull's-eye moves down beside the letter C. When you leave the menu later, the clock will no longer be displayed. It will continue to keep time, however, so you'll be able to retrieve it later with menu choice B.

Set Clock (D): When you first access the menu, the clock at the top probably won't be running. If you press D, the cursor will appear at the time display. Simply type in the present time directly over the display, making sure to set AM or PM, then press the return key. The clock will be set to whatever is showing in the time display area and will start running immediately.

To get a stopwatch/timer, select D, then press shift-CLR to clear the screen. When you press the return key, the stopwatch, set to zeros, begins running. Press D again to stop it.

Set Alarm (E): The alarm time is shown at the bottom of the menu. When you press E, the cursor appears there, and you enter the alarm time just as you did when setting the clock, again remembering to set AM or PM. When you press the return key, the alarm is enabled, and the display changes to AL ON to remind you.

When the preset time arrives, a buzzer or bell will sound, the screen will flash orange and black, and ALARM will be printed on the screen. You acknowledge it by pressing the shift and control keys simultaneously, which, of course, brings up the menu. If the alarm sounds while the menu is displayed, you can cancel it by pressing F.

No Alarm (F): This option disables the alarm and changes the display to AL OFF. The time set remains displayed, however, and you can reactivate it later by pressing E and the return key (to enter the time shown). This would be handy if, for example, you were using the computer alarm to get up for work or school and

wanted to disable it temporarily on the weekend.

Print Screen (G): Pressing G sends whatever was on the screen to the printer. This is a simple routine, so don't expect anything fancy. It's great for a quick snapshot of the screen, though.

Print Memo (H): Option H sends the contents of your memo pad to the printer.

HOW IT WORKS

Because the program is in machine language, you can achieve some effects—such as the instant transfer of information screens—that are almost impossible to achieve in Basic. The program for the C-64 is placed in high RAM at 49152, and storage for extra screens of information is in hidden RAM behind the Basic ROM. The C-128 version is located in the applications program area at RAM location 4864, and extra screens are stored at the top of Bank 0, slightly reducing the space available for Basic programs.

The program also uses the Interrupt routine of the C-64 or C-128. Sixty times every second, the computer stops what it's doing, sets aside its work and performs some "housekeeping"—reading the keyboard, updating the TI$ clock, and so on. By modifying the vector for this Interrupt routine so that it points to your program, you can add the extra functions to the list of housekeeping chores. This allows you to carry out other activities, such as writing or running another program, while Handy Window checks in to see whether it's needed. In effect, the computer is running two programs simultaneously.

The digital clock doesn't use TI$ to get the time because TI$ is inaccurate, particularly on the C-64. It tends to run too fast and stops entirely when the computer accesses a peripheral. Instead, the clock is based on the time-of-day feature of one of the two 6526 CIA interface chips. These chips are used for timing a variety of internal activities, as well as communicating with the disk drive and modem. The time-of-day feature is not normally used, but it's extremely accurate. Because there are two CIA chips, you could theoretically have two clocks for different time zones.

FINAL TIPS

Handy Window shares computer assets with whatever other programs you might be running. It will do just fine as long as the other programs don't overwrite the same areas in memory. But it won't work if the other programs change the interrupt vector, and it's incompatible with any program that uses the same RAM

for storage. If you're uncertain whether a program is compatible with Handy Window, you'll have to try it and find out.

Don't attempt to use Handy Window when any input/output processes (such as printing, disk access or modem communication) are active. Because Handy Window interrupts normal operation, it will shut down these activities automatically.

If the Handy Window menu should fail to appear when you press the shifted control key, the interrupt vector probably has been changed. This can happen easily—for example, if you press the run-stop/restore combination by mistake. To restore Handy Window, just type

```
SYS 49152 <return>
```

for the C-64, or

```
SYS 4864 <return>
```

for the C-128.

Finally, Handy Window doesn't work on the C-128's 80-column screen. If you invoke it in this mode, you'll get the menu, but none of the functions will work properly. ■

Basic Bug Trap

By Michael Broussard

RUN It Right

C-64

If you're cursing the elusiveness of the bugs plaguing your programs, Debugger 64 can come to the rescue! This utility eliminates the need for adding and deleting Stop and Print statements to find a Basic program bug. Instead, it adds to Basic several new commands that let you set dynamic breakpoints, step through your program one statement at a time, examine your program's history of Gosub invocations and view the line number of each statement as it's executed.

To use Debugger 64, load it into memory and run it to activate the debugging commands. Then load the program you want to debug, or type one in from scratch.

Debugger 64's commands are meant to be used in Direct mode, but most will also work from within another program. The commands begin with the @ character, to distinguish them from regular Basic commands.

A brief description of each command appears in Table 1. Their use is demonstrated by the sample Basic program below.

DEBUGGER 64 IN ACTION

Load and run Debugger 64, then type in and save the following code:

```
10 PRINT "SAMPLE DEBUGGER
   PROGRAM"
20 GOSUB 100
30 PRINT "END OF PROGRAM"
40 END
100 PRINT"THIS IS LINE 100.":
    PRINT"TIME TO GOSUB!"
110 GOSUB 200
120 RETURN
200 PRINT"THIS IS LINE 200."
210 GOSUB 300
220 RETURN
300 PRINT "THIS IS LINE 300."
310 X=10
320 X=X*2+5 : RETURN
```

When you're done, run the sample program once to see how it works. Then enter @HELP or @? to view a summary of the Debugger 64 commands.

```
*** DEBUGGER COMMAND SUMMARY ***

@TRON             -- TRACE ON
@TROFF            -- TRACE OFF
@BREAK <LINE>     -- SET A BREAK
@LIST             -- LIST BREAKS
@RESET <LINE>     -- RESET 1 BREAK
@RESET *          -- REST ALL BREAKS
@CONT             -- RESUME EXECUTION
@STEP             -- STEP 1 STATEMENT
@STACK            -- GOSUB STACK TRACE
@OFF              -- TURN OFF DEBUGGER
@HELP, @?         -- COMMAND SUMMARY
```

Now enter @TRON and run the sample program again. Note how Debugger 64 prints a line number message in reverse video before each line of the sample is executed.

Next turn @TRON mode off by entering @TROFF, set a breakpoint by entering @BREAK 200 and run the sample program a third time. Execution pauses just before line 200, and a Break in 200 message appears.

To examine the history of Gosub invocations, enter @STACK. The resulting display looks like:

```
*** STACK TRACE ***
GOSUB 100 FROM LINE 20
GOSUB 200 FROM LINE 110
```

Enter @BREAK 300 and press the return key, followed by @BREAK 310 and another return. Then, to check the breakpoints you've set, enter @LIST. Three breakpoint numbers—200, 300 and 310—should appear.

Now resume execution of your program by entering @CONT. The Print statement in line 200 will be executed, but the next breakpoint will make the program halt at line 300. Enter @CONT once more to continue execution until the last breakpoint produces the message, Break in 310.

Then, when you reinspect the Gosub history with @STACK, the following display appears:

```
*** STACK TRACE ***
GOSUB 100 FROM LINE 20
GOSUB 200 FROM LINE 110
GOSUB 300 FROM LINE 210
```

@HELP or @?: Displays a summary of all the Debugger 64 commands on the screen.

@TRON: Trace On. Displays each line number in reverse video before the line is executed, so you can trace the flow of your program.

@TROFF: Trace Off. Disables the line number tracing activated by @TRON.

@BREAK: Sets a program breakpoint. This command must be executed in Direct mode, and it must be followed by a line number.

For example, if you enter @BREAK 235, Debugger 64 will suspend execution before each statement in line 235 of your program. Then a break message will tell you at what line the break occurred. When your program has stopped at a breakpoint, you can list portions of it, display or change the values of variables, set more breakpoints, and so on.

@BREAK is an improvement over the standard Basic Stop command, because it lets you set breakpoints dynamically without altering your program.

@CONT: Resumes execution from a breakpoint. Don't try to use the Basic CONT command to continue execution from a Debugger 64 breakpoint! Similarly, executing @CONT when your program is not at a debugger breakpoint has unpredictable results.

As with the Basic CONT command, if you alter your program or encounter an error while execution is suspended, you can't continue execution, but must restart the program with the Run command.

@STEP: An alternative to @CONT. When your program is at a breakpoint, @STEP performs the next logical program statement, then suspends execution as if another @BREAK had been encountered. In this way, you can execute one statement at a time without having to set breakpoints at every line. Since @STEP is an alternative to @CONT, use it only when your program is stopped at a debugger breakpoint.

@LIST: Displays a list of all the breakpoints currently set.

@RESET: Clears a breakpoint. For example, @RESET 240 removes the breakpoint from line 240. Enter @RESET * to reset all the breakpoints at once.

@STACK: Displays all currently active Gosub invocations. If more than one Gosub is active on the stack, they're listed in order from the first to the most recent.

@OFF: Disables Debugger 64. After using this command, debugger commands won't work until you re-enable them by typing SYS 49152.

Table 1. Debugger 64 commands.

Examine line 310 by entering LIST 310, and check the value of X by entering PRINT X. Since line 310 hasn't been executed, the value should be 0. Once you've verified this, enter @STEP. Debugger 64 then executes line 310 and prints a Stepping: Break in 320 message to the screen.

Next enter PRINT X again and notice that the value of X has changed to 10. Enter @STEP and inspect the value of X once more; this time it should be 25. Finally, resume execution of your program with @CONT.

When the End of Program message appears, enter @RESET * to delete the breakpoints. Then use @LIST again to verify that they no longer exist.

ADDING COMMANDS TO BASIC

Each time the C-64 operating system executes a Basic program statement, it jumps to the address stored in RAM locations 776 and 777. Debugger 64 changes the addresses in these locations so the operating system jumps to the debugger, rather than to the normal Kernal ROM, to execute a statement.

When in use, Debugger 64 checks to see if the next statement is a debugger command and, if so, executes it. If not, it jumps to the Kernal ROM so that the statement is executed normally by Basic. ■

Sprite-Print

By Roy Duncan

RUN It Right

C-64; printer

Movable object blocks, better known as sprites, are a nice complement to the C-64's graphics. Unfortunately, sprites can't be printed in a conventional manner. This program is for those of you who have always wanted to print sprites but were afraid to ask how.

Sprite-Print is a menu-driven program that lets you copy sprites onto a high-resolution graphics screen beginning at address 8192. The program prints the contents of this hi-res screen and also allows you to copy and print text along with the sprites.

HOW TO USE SPRITE-PRINT

Sprint-Print 1 is the main program. It does all the work, with the exception of printing the screen. Sprint-Print 2 prints the screen by creating a short machine language file called PDUMP/1525. Whenever you select the Print option, this file is loaded from the disk. Sprite-Print 3 creates the sprite.

First, run Sprite-Print 3. When the cursor reappears, load and run Sprite-Print 1 without clearing the screen. When the options screen appears, select option 6 to clear the hi-res screen, then press 1 to create a screen in a different area of memory. Next, select option 2 or 3 for uppercase or lowercase, respectively. Finally, select the normal Commodore character set.

After a few minutes, the sprite created by Sprite-Print 3 will begin to form on the screen. Once the cursor returns, choose option 4 from the main menu to print the sprite. When the program is finished, press any key to return to the options screen.

PROGRAM OPERATION

Here's how each option in Sprite-Print 1 works:

Option 1, Create Screen. When you select this option, the computer asks what you want to copy, then places that data on

the hi-res screen. This option does not alter the contents of the screen. If you want to copy sprites, you may not want to erase the hi-res screen.

Option 1 also offers the ability to load an alternate character set from disk. The first two bytes of the character-set file contain the address from which that file was saved. Since these bytes are not part of the character set, they are removed in line 2590. If your character set appears to be altered, change line 2590 to:

```
2590 T=50176
```

Option 2, Save Screen to Disk. This option saves a hi-res screen to disk. To see how this feature works, run Sprite-Print 3, then run Sprite-Print 1 without clearing the screen. Select option 6 to clear the hi-res screen, followed by option 1 to create a screen. Follow the prompts generated by option 1 to create your sprite on the hi-res screen. Once the sprite is created, press any key to return to the main menu. Finally, select option 2. Enter a filename at the prompt and press the return key.

The hi-res screen is saved to disk as a program file. If you choose this option accidentally, enter filename $ to return to the main menu.

Option 3, Load Screen from Disk. After you choose this option, follow the steps outlined for option 2 to create a screen file. Place the disk containing the screen in the drive and select option 3. Enter the screen's filename and press the return key to load the screen into memory. The program will remain at the main menu if it encounters a load error. Again, if you should happen to choose this option accidentally, enter filename $ to return to the main menu.

Option 4, Print Screen. This option prints the graphics screen in memory to a 1525, MPS-801, MPS-803 or other 1525-compatible printer. When you call this routine, the machine language file created by Sprite-Print 2, called PDUMP/1525, is loaded from disk. If a disk error occurs during loading, the program will return you to the main menu. You can change the PDUMP/1525 filename as long as the change is reflected in the value F$ in line 2260.

Option 5, Display Screen. This displays the hi-res screen in memory. Pressing any key returns you to the main menu.

Option 6, Clear Screen. This option clears hi-res color memory and the hi-res screen.

Option 7, Exit Program.

When the program is waiting for you to press a key, the cursor will flash in the upper-left corner of the screen. This tells you that

the program has finished drawing the hi-res screen.

SUBROUTINES

Sprite-Print also contains three short machine-code subroutines. The first, accessed with SYS 828, copies 128 characters from ROM to RAM. The second is activated by SYS 869 and clears hi-res memory by poking zeros into locations 8192 to 16191. The third subroutine is activated with a SYS 923 and fills hi-res color memory, beginning at 1024, with predetermined values. The default color value is 12, which creates black plotting on a medium gray background. To change the color combination, change the POKE 251,12 in lines 80 and 2340 to POKE 251,x, where x = [plot color]*16 + [screen color].

THE PRINT SUBROUTINE

I designed the printing subroutine as a separate program so you can use it in your own programs. Just load the subroutine with the command

```
LOAD "filename",8,1
```

Then poke 49152 with the value of the hi-res screen's starting address divided by 256. Sprite-Print prints the hi-res screen beginning at 8192, requiring you to enter

```
POKE 49152,8192/256
```

before printing. The Print subroutine can be executed as follows:

```
OPEN4,4:PRINT#4:SYS 32959:
PRINT#4:CLOSE 4
```

The routine uses locations 251 to 254, as well as 49152 to 49173.

Whether you use Sprite-Print in its entirety or use just a few of its subroutines, I think you will find it useful for many applications. ■

PrintRite 64

By Bob Kodadek

RUN It Right

C-64; 1525, MPS-801 or MPS-803 printer

From your experience with typewritten or printed material, you know that some lowercase characters (g, j, p, q and y), the comma and the semicolon have "tails" extending below the base of the printed line. These are called descenders.

Unfortunately, some Commodore dot matrix printers, such as the 1525, MPS-801 and MPS-803, use fonts in which the punctuation marks and characters with descenders are moved up above the printed line, making them look unnatural and sometimes difficult to read. Although adequate for rough drafts, this type of print is unsuitable for formal documents or correspondence. These printers also lack a much-needed underline command.

PrintRite 64 is a handy C-64 wedge program that adds true descenders to the lowercase letters and punctuation produced by the 1525 and MPS-801/803 printers. It also creates the missing Underline command.

As an added bonus, PrintRite 64 is designed to boot RUN Script 64 Version 1.0 (*RUN*, March and April 1986). It can also function in Immediate mode and be accessed by any Basic or machine language program as long as certain command restrictions are observed.

USING PRINTRITE 64 WITH RUN SCRIPT 1.0

Any time you wish to use PrintRite 64, just load and run it. The program will prompt you with LOAD RUNSCRIPT? (Y or N). If you answer yes, the program asks for the device number. You then insert a disk or tape containing RUN Script and press 1, 8 or 9, as appropriate. The program searches for and loads a program file named RUNSCRIPT. It then moves the character set, defines the macros, alters the vectors and boots RUN Script.

Before printing a document, you must change the page length from the default value of

66 to a value of 49. Enter the RUN Script dot command .pl49 at the beginning of each document to make this change.

To underline a word, sentence or phrase, use the macro character u. Place this macro before and after the text to be underlined. Other predefined macro characters are R (reverse on), r (reverse off), D (double width on), and d (double width off).

COMMAND LIMITATIONS

To preserve the integrity of PrintRite 64, the only valid print commands are as follows:

CHR$(13)—carriage return
CHR$(14)—enter Double-Width mode
CHR$(15)—enter Standard Character mode
CHR$(17)—underline toggle
CHR$(18)—enter reverse field
CHR$(146)—turn off reverse field

Further restrictions apply to the use of double-width or reverse-field characters. In neither case can you include any special characters in the word or phrase to be printed, and any double-width text must appear on a separate line.

RUN Script 1.0 normally uses all the C-64's memory by switching out the Basic interpreter and leaving only the operating system, or Kernal, intact. RUN Script uses locations 6691 and 6692 as pointers to the top of available memory. The Boot routine places a smaller value in these two locations to allow PrintRite 64 to reside concurrently in RAM at 51968–53247.

USING THE PROGRAM FROM BASIC OR IMMEDIATE MODE

To use the wedge in Immediate mode or from a Basic program, first load and run PrintRite 64. When prompted to load RUN Script 1.0, press N. Always open the printer with a device number of 4 and secondary address of 7; for example: OPEN 4,4,7. Any program that observes the above guidelines and doesn't occupy the same memory area should be compatible with PrintRite 64.

You can disable the wedge from Basic by pressing run-stop /restore from Basic or after exiting RUN Script. You can restart it by entering SYS 52296.

The printer's default comma and semicolon are used to increase output speed. If you want to use these characters, remove the two REM statements in lines 10 and 11 of the Basic loader before running it. You can also make these two Pokes in Immediate mode. To disable either character, poke a 10 in the location shown.

HOW IT WORKS

All data transferred for output

on the computer's serial bus is intercepted and handled by a wedge at locations $0326–$0327 (806–807). Location $9A (154) is checked to see whether a channel to a printer was opened with device number 4. If this value is not found, the data is sent to its intended destination—the screen or the disk drive.

If a channel to a printer was opened, the data is tested to see whether it's a command being sent to the printer. All commands other than CHR$(13) and CHR$(17) are validated and sent to the printer. The remaining bytes are considered to be characters and are then evaluated as either standard or special. Special characters are the lowercase letters g, j, p, q and y, along with the comma, semicolon, quotation mark and underline. All standard characters are printed using the character set residing in the printer's ROM. Special characters, printed in Graphics mode, are really composed of two characters, one the top half, and the other the part that descends below the line.

When PrintRite 64 is activated, the printer prints each line in two passes. On the first pass, the routine counts the leading spaces and determines the left margin. It then positions the printhead, using the command to set a tab, CHR$(16). The program prints all normal characters and the top half of all special characters on this pass.

On the second pass, the program checks the buffer for special characters, underline toggles and carriage returns. PrintRite then determines the placement of these special characters. It locates the printhead, looks up the graphics data from a table and prints the character.

To speed up the printing process, the routine scans from the current print position to the end of the line. If no valid characters exist, or if the line consists of spaces, a carriage return is printed immediately. Without this technique, printing speed would be reduced to a crawl.

All carriage returns are printed in the Graphics mode at nine lines per inch. This technique allows no spacing between the top and bottom of each character. Therefore, each two-part special character appears as a unity. Since two carriage returns are required for each printed line of text, the line-feed spacing effectively becomes 4½ lines per inch. This is why the page length must be 49 instead of the normal 66.

PrintRite 64 can improve the quality of your documents, making them more readable and attractive. It's a welcome addition to any word processing application. ■

Disk Stuffer

By Steve Canny

RUN It Right

C-64; C-128; 1541 disk drive, 1571 disk drive (in 1541 mode)

Long Directory is a group of three programs that utilize normally wasted disk space when you're storing lots of little graphics files, such as those created by The Print Shop and PrintMaster. It will let you save up to 288 of these files on one side of a 1541 disk, provided they don't exceed a total of 664 blocks.

To create a "long directory," load and run Long Directory1, remove the program disk, insert a blank data disk and follow the screen instructions. Long Directory1 allocates track 19 on the data disk, so the DOS won't save any files there.

When Long Directory1 is finished, test it by entering the New command, followed by this one-liner:

```
10 FORT=1TO144:SAVE"TEST"
   +STR$(T),8:NEXT
```

This will save 144 test files to the data disk in about ten minutes. Then load and run Long Directory2 from the program disk, reinsert the data disk and follow the screen instructions again. Long Directory2 transfers the contents of track 18 to track 19 and reformats track 18 to accept an additional 144 files. When Long Directory2 is done, enter the New command and use the one-liner to add another 144 test files to the data disk. Just change the TEST in the one-liner to TESTS this time around.

Now load and run Long Directory3 from the program disk, reinsert the data disk and follow the screen prompts. Long Directory3 finds the last sector on track 18 on the data disk and changes the sector link to point to sector 1 on track 19, which contains the second directory added to the disk. Long Directory3 then links the first 144 files with the second 144 files and validates the disk to prevent overwriting. After you've modified the data disk with Long

Directory3, place a write-protect tab on it to prevent alterations.

You can't validate disks modified by Long Directory, nor can you scratch files from them or add files to them. Also, if you have a 1571 drive, you can read the directory in 1541 mode only.

When you load the data disk's directory and list it, you'll find that the test file appears 288 times. To use Long Directory with your own files, just substitute them for the one-liner after running Long Directories1 and -2. ■

64 Notepad Command Center

By Bob Kodadek

RUN It Right

C-64; printer optional

In the January 1987 issue of *RUN*, I presented the first enhancement to my 64 Notepad program from the September 1986 issue. For those who may have missed it, 64 Notepad is a program that takes over control of the C-64 to provide an instant-access text window for jotting down and recalling programming notes—without affecting the original screen display. It is a machine language routine that's transparent to most other Basic programs, and so won't interfere with them.

The program in this article, a Basic loader called Notepad DOS, will add even more power to your 64 Notepad window. It provides a menu-driven DOS command center for viewing the disk directory, formatting or validating a disk, initializing the drive, or renaming, copying or scratching files, each operation accomplished by pressing only one key. I've also included a command that loads a binary file without altering the Basic memory pointers.

Because you don't need to enter command strings, Notepad DOS is faster, more convenient and easier to use than the DOS Wedge. Its functions will work while another program is executing, and you'll find them self-explanatory. The Simple Append program that was employed in January to create 64 Notepad II is also used here to create Notepad III, by adding Notepad DOS to Notepad II.

Since the Notepad DOS enhancement makes 64 Notepad rather large, it now takes a long while for the Data statement information to be poked into memory. To save time, I've included a small Basic program, called Notepad Save, that will

D	Directory
N	Format
R	Rename
I	Initialize
B	BLoad
V	Validate
C	Copy
S	Scratch
Q	Quit

Table 1. Notepad DOS commands.

To obtain the 64 Notepad, 64 Notepad II and Simple Append programs, order the September-October 1986 and January-February 1987 ReRUN disks. You'll find an order form on the back page of this booklet.

make a fast-loading binary file out of Notepad III.

CREATING NOTEPAD III

First, you must have a working copy of 64 Notepad II from January on disk. [If you don't already have the 64 Notepad, 64 Notepad II and Simple Append programs, see the note in the table above for instructions on obtaining them.] Then load Notepad DOS, so you can add it to Notepad II with the Simple Append program. The proper syntax for an append is SYS (SA), "FILENAME", with the variable SA holding the starting address of the append routine. The SA value you should use is the default address 828 in the cassette buffer.

Load and run Simple Append, then load Notepad II. Next, place the disk containing Notepad DOS in the drive, and, in Direct mode, enter the following command:

```
SYS 828,"NOTEPAD DOS"
```

If no errors occur, the OK message will appear, indicating that you have a copy of Notepad III in memory. To save it, enter the following line:

```
SAVE"64 NOTEPAD III",8
```

Now, with the program safely on disk, you're ready to try it out.

USING THE NEW ROUTINES

You can access the DOS menu whenever the Notepad window is open by pressing control/D. The one-key commands that will appear are listed in Table 1. To select the function you want, just press the appropriate key. If the routine needs any information from you, it will issue a prompt at the appropriate time.

You can abort any operation on the menu by pressing the stop key, and the current status of the error channel always appears at the bottom of the menu screen. To return to viewing your notes,

select the Exit option, or, to return to Basic, press control/C to close the window.

CONVERTING BASIC TO BINARY

As I mentioned, I've included in the Notepad package a program called Notepad Save for converting Notepad III into a fast-loading binary file. It would be a good idea to store the binary file that Notepad Save will create on a newly formatted disk, so prepare one and have it handy. Then turn your computer off and on to clear out any resident programs.

Next, load and run Notepad III, and, at the ready prompt, press the run-stop and restore keys simultaneously to disable the Notepad interrupt. Now, load and run Notepad Save, and when it asks for the filename and locations of the machine language program, respond as follows:

```
FILENAME? 64 NOTEPAD V3.0
START ADDRESS? 50448
END ADDRESS? 52801
```

When you insert the formatted disk and press return, Notepad will be stored as a binary file.

To execute this file, you need the Notepad Boot program. Save it on the same disk as the binary file. Now, whenever you want to use 64 Notepad, just load and run Notepad Boot. The binary file will boot in only eight seconds, a considerable saving in time.

You can use Notepad Save to make a binary file out of any data area in memory, whether it be a machine language program, a hires screen, a sprite or something else. All you need are the starting and ending addresses of the section of memory you want to be saved. These binary files can be loaded with the ,8,1 syntax or the Notepad DOS BLoad command. ■

Power Key

By John Ryan

RUN It Right

C-64

Like many other C-64 owners, my collection of programming utilities has grown considerably over the years. Unfortunately, most of my Basic utilities are not tailored to meet my specific needs; I'm left at the mercy of the author's own idea of what a utility should and should not do.

I wrote the Power Key program to help me design a utility that meets *my* needs, not those of someone else. It provides one-key entry of Basic keywords, plus other commands that perform various screen and editing functions.

Power Key lets you assign up to 16 Basic keywords to the function keys on your C-64; or you can choose the program's Default option, which assigns predefined keywords to the function keys. As a bonus, a number of functions that enhance the control key are also included.

USING POWER KEY

After loading Power Key, type SYS 49152 to run the editor portion of the program, where you can customize it as you wish. You're given the two options of assigning the default keywords or your own.

If you choose the Default option, Power Key assigns the two sets of Basic keywords in Table 1 to the function keys. If you choose to define your own, Power Key prints the name of the function key and prompts you to enter a Basic keyword. Since Basic 2.0 keywords can be no more than six characters in length, Power Key reads only the first six characters of your input.

To keep the program short, I've included no check for spelling, so be sure that each keyword you enter is spelled correctly. Otherwise, it will cause a syntax error when used in your program.

Once you've assigned the keywords, you can save them to disk as a small machine lan-

guage file under a filename of your choice. After saving the file, you're given the option to create another file. If you answer yes, the process of assigning keywords repeats. If you answer no, the program exits to Basic. Should you choose to create several files with many different keywords, I suggest writing the filenames and their function key definitions on 5-by-7 cards for reference.

PROGRAMMING WITH POWER KEY

Using Power Key in your own Basic programming is easy. Don't load the main Power Key program. Instead, load one of the files you've created with the program. Use the syntax LOAD "filename",8,1 for this operation.

After the file has loaded, type NEW to reset the Basic pointers, and enter SYS 49662. Power Key will then be activated, along with the following key functions:

F1–F8: Print Basic keywords.

Commodore key: Toggles the computer between the two sets of Basic keywords. When you initially activate a Power Key file, keyword set #1 is active. Press the Commodore key to access the second set, and press it again to change back to the first.

Control/B: Changes the border color.

Control/S: Changes the screen color.

Control/L: Changes the color of every character on the screen.

Control/E: Escapes Quote mode. Control/E is the only function that works while the computer is in Quote mode, deactivating it if you need to use Power Key functions. It provides a way to enter keywords assigned to the F1–F8 keys and control/color combinations within Basic strings.

Holding down the control key: Freezes the C-64's time clock and suspends the execution of a Basic or machine language program until the key is released. This function is useful for examining Basic program listings, or for examining screen displays during a program's execution. The Commodore key can be used in a similar manner, but it toggles the keyword set each time it's pressed.

To exit Power Key, press the run-stop/restore key combination.

All of Power Key's functions work in both Program and Direct modes. This is possible because the program diverts the hardware interrupt vector (CINV $0314–$0315) to point to the Power Key routines before performing housekeeping chores like flashing the cursor, updating the clock and scanning the keyboard.

If a function key has been

SET #1	SET #2
F1 LIST	F1 DATA
F2 GOTO	F2 READ
F3 GOSUB	F3 THEN
F4 RETURN	F4 NEXT
F5 PRINT	F5 RIGHT$
F6 REM:	F6 LEFT$
F7 POKE	F7 LOAD
F8 PEEK	F8 SAVE

Table 1. Lists of default keywords.

pressed, the program attempts to match the key's ASCII value with values stored in a reference table in memory. When a match is found, a Basic keyword is printed; otherwise normal program control continues.

If you've pressed the control or Commodore key at location 653, Power Key either performs a screen function or toggles the keyword set, depending on which key you pressed. A value of 2 represents the Commodore key; a value of 4, the control key. You can include the statement X = PEEK (653):PRINT X in your Basic programs to determine if either the shift, control or Commodore key has been pressed. ■

Drummodore

By Larry Cotton

RUN It Right

C-64

Drum machines, digital devices that simulate the sounds of percussion instruments, are fast becoming popular with both professional and amateur musicians. The sounds are encoded on ROM chips, so the "drummer" has only to tap a few keys on the keyboard to sound like a studio musician.

I wrote Drum Machine to emulate commercial drum machines. It demonstrates basic features of the more sophisticated products and will let you create some drum music of your own. With it, you can record and play sequences of drum beats, edit them, vary their tempos, place them in continuous loops and save them to and load them from disk.

Although the program is written completely in Basic, it can play a sequence accurately and quickly. I achieved the speed by assigning letter names to constants and by keeping the main recording and playing loop compact. I used the C-64's built-in TI function (not For/Next loops) to create accurate beat durations.

THE MENU

When you run Drum Machine, a menu listing all the sounds the program will make, plus the program functions, will appear on the screen. To produce a sound or access a function, just press the appropriate key.

The first group of menu items includes the nine percussion sounds and an asterisk. The asterisk is used to define loop sequences, as I will describe later.

Summaries of the function key commands appear next. There are nine of them, instead of the eight you might expect, because F5 toggles between the Play and Stop modes.

The return key erases the sequence you're working on so you can start over from scratch. This function includes a safety message, so if you hit return by mistake or change your mind

about starting over, you can stop the erasure.

The next two items on the menu screen let you change the tempo of a sequence and choose the program mode you want to work in. One more item, which counts the number of beats in a sequence, appears on the menu screen when you access Record or Play mode.

PRACTICING

All the sounds are produced with eight of the bottom-row keys and the space bar. You'll find it easiest to play with the fingers of your left hand on keys Z through V, the fingers of your right hand on B through < and your thumbs on the space bar.

The program starts off in Practice mode, where the sounds don't get recorded. Practice awhile to get used to the sounds and the almost imperceptible lag between pressing a key and hearing it. When you're recording, you'll have to ignore this lag, because if you wait for the computer to "catch up," you'll find yourself playing slower and slower.

Stay in Practice mode until you get the sequence the way you want it; then press F1 to record.

RECORDING

In Record mode, when you press the first sound key, the timer will start and keep running as long as you're in that mode. Because it keeps running, if you pause after a beat, the computer will think that beat is a very long one. Then, when you play the sequence back, the computer will seem to get stuck, producing no sounds, while it waits for the "long" beat to end. To fix a situation like this, you have to edit the long beat out; see the editing instructions below.

In contrast to some professional drum machines, my program can produce only one sound at a time, and there's no provision for varying the duration of a beat. The maximum number of beats you can record is 500, but there's no signal when you pass the limit.

To increase or decrease the tempo of a sequence, press F3 or F4, keeping in mind that the beats will start to run together at about 30. It's best to record at tempo 10. If you have a difficult sequence to record, try pressing the keys slowly, then speeding up the tempo, using F3.

To listen to your sequence at any point, press F5, for Play mode. After the entire sequence has played through, the computer will automatically switch back to Record mode, where you can add more beats or edit what you've already got.

You can halt a playing se-

quence by tapping F5 again to place the computer in Stop mode. Then, from Stop mode, you can access other functions by pressing the appropriate key, or return to playing—from the beginning of the sequence—by pressing F5 yet again. Any sounds you play in Stop mode won't be recorded.

Pressing F7 or F8 lets you load a sequence from disk or save it to disk. When loading, be sure to type the sequence's name exactly as you saved it.

EDITING

To edit a sequence, you must be in either Record or Stop mode. The only way to make changes is to press F6 to back up through the sequence, erasing beats as you go. Then you can add on replacements as you wish.

Because this is Drum Machine's only editing feature, if you want to make a change near the beginning of a long sequence, it's easiest to just hit return and start over. It's even better to edit a sequence in small chunks as you go along, so you don't have to make changes a long way back.

DEFINING LOOPS

You can use Drum Machine to accompany other music by placing a sequence in a continuous loop. You do this in Record mode by pressing the * key after the last beat you want in the loop. Be sure to allow an instant of time between the last beat and hitting the * key, so you don't cut the last beat short.

Once you've flagged the end of the loop with an *, the only way to add to it is to edit out the flag. Probably the minimum you'd want for a continuous loop sequence would be one bass drum beat and one snare drum beat. ■

Robotgraph

By Rick Kephart

RUN It Right

C-64

Robotgraph is a "multimedia" program I designed to help children in the middle grades learn how to use bar graphs. The program's animation will capture the youngsters' attention, and the sound will reinforce the relationship between the value of each bar and its height.

When you run Robotgraph, it first sets up a horizontal x-axis, labeled A through H, and a vertical y-axis, labeled 1 through 10. The bars, A–H, will extend up from the x-axis to values measured on the y-axis.

Next, the program requests a value for the first bar, A. You can enter any whole number from 0 through 10 or any half-value in between, written in decimal form.

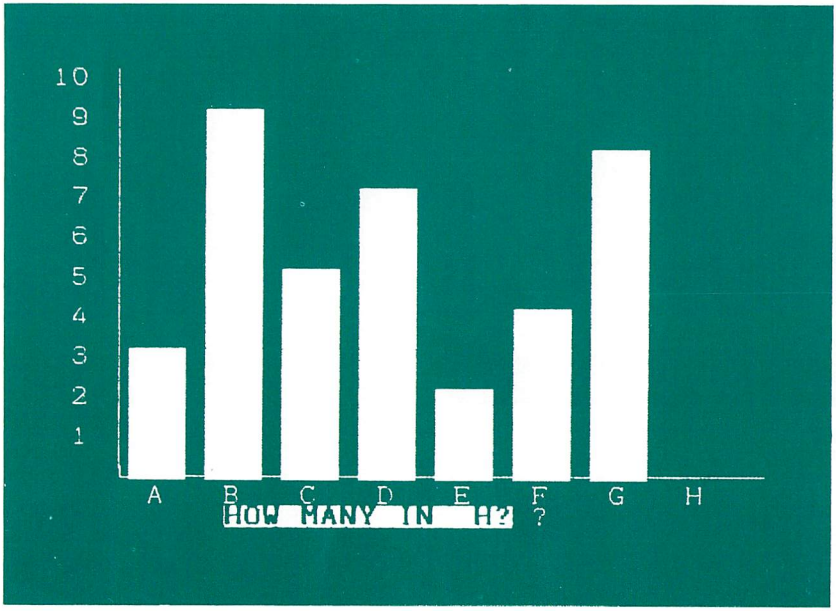
After you enter a valid number, a beeping robot descends from the top of the screen, and as it reaches the x-axis, it stops and extends a claw. The robot uses the claw to "lift" the bar representing the value you typed in up to the correct height. The growth of the bar is marked by a tone at each increment of one.

When the bar is complete, the robot retracts its claw and disappears off the top of the screen. Then Robotgraph requests a value for the next bar and repeats the process until eight different colored bars have been drawn.

You can review the value of each bar by pressing the letter corresponding to that bar. Pressing the return key will set up a new graph.

To examine the sprites responsible for the animation, use the following procedure. First, load and run the program normally, inputting any valid value for A. As soon as the robot is about halfway down the screen, hit the run-stop key to halt execution. (Don't hit the run-stop and restore keys together.)

Now press shift/clr-home to clear the screen of all but the robot, and, with the cursor at the top of the screen, type POKE B, followed by



a number from 200 through 210 (the block numbers). POKE B,200 and POKE B,201 will display the two sprites used for the descent and ascent animation. POKE B,202 through POKE B,209 will display the sprites in the arm-out animation (or arm-in, in the reverse order). POKE B,209 and POKE B,210 will display the two sprites used for the ascend-with-arm-out animation.■

Mega-Magic—May

By Jim Borden

RUN It Right

C-128 (in C-64 mode)

The C-128 Keys in C-64 Mode program will let you use your C-128's keypad, no-scroll key and cursor keys while running programs in C-64 mode. It works by rewriting the IRQ routine so new keyboard-scan code can be added. The program stores itself at addresses 49152–49454, but, if you're familiar with machine language and have need, you should be able to move it elsewhere with little trouble.

The program activates the no-scroll key by changing the Kernal CHROUT vector to freeze the computer in the IRQ routine when the key is pressed. When you've paused printing, you can press any key to start it again, but I'd suggest using the no-scroll key.

The 128 Keys in 64 Mode program may not work with some commercial software, depending on how much the latter changes the Basic pointers. However, it should work with any Basic program that doesn't use the same memory addresses.■

Mega-Magic—June

By Jim Borden

RUN It Right

C-64; C-128

The Shape of Sprites to Come is a program that cycles through the creation and deletion of four sprites stored in an array.

When you run the program, the first sprite that appears replaces its bytes until it assumes another shape. Delete lines 80 and 90 to see the order in which the bytes are replaced. A random number is swapped with the *X*th element in the array, to shuffle the order in which the image changes, with only one pass through the array.

The Pokes in line 100 determine the screen location of the sprite (for the C-64) and expand the sprite in two directions. Expanding makes it easier to see the sprite as its bytes are replaced. Line 110 sets the sprite block to the proper location, turns the sprite on and sets the sprite's color to white.

The value of Y cycles through the sprite images in line 120. The For-Next loop takes the *X*th number from the order array, adds it to the sprite memory pointer and pokes that location with the appropriate byte from sprite image Y. Sprite image Y is then incremented and the range checked. Have fun experimenting with different values in all of these locations.■

Puzzler's Choice

By Daniel Miller

RUN It Right

C64

You are hereby invited to try Puzzler's Choice, which contains three puzzles for your entertainment.

ALPHABET PUZZLE

Alphabet Puzzle is the microcomputer rendition of the move-the-blocks puzzle we've all played at one time or another. In this version, the tiles originally numbered 1–15 are lettered A–O. Your objective is to arrange the letters into the sequence printed on the screen.

Enter the letter of the tile you want to move, and, if the move is valid, the tile will enter the empty yellow space. Illegal moves make an error message flash on the screen, after which you can re-enter your move.

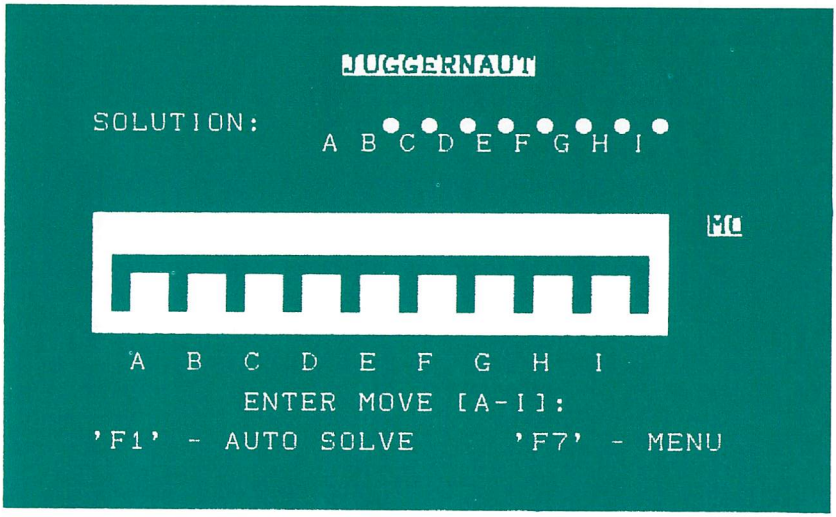
There are more than ten trillion solvable combinations for this puzzle, with an equal number that are impossible to solve. Don't worry, though—the computer will only present possible combinations. At times, you may see the puzzle board change several times until the program settles on a solvable arrangement.

JUGGERNAUT

Juggernaut uses sprites and keyboard graphics to provide another challenge. The objective is to rearrange the alternating sequence of red and yellow "checkers" into a line of yellow checkers on the left and red checkers on the right.

Beneath the playing board appear the letters A–H. Select a letter, and the two checkers above it will rise into the empty slots. Consider your moves carefully, because you get only four of them. If you get stuck, press F1 to initiate the Auto-Solve mode.

As to the name of the puzzle, the playing board fancifully resembles the multiwheeled cart, the Juggernaut, that carried the incarnation of the Hindu deity, Vishnu. Frenzied worshippers threw themselves beneath the wheels as it passed by.



CHECKERS SWITCH

I consider Checkers Switch the toughest challenge of the three puzzles in this collection. Red and black checkers are arranged on opposite sides of the checker board. Using only legal checkers moves and jumps, you must switch the positions of the checkers to the opposite sides.

Enter your moves by designating the row and column of each space you want to move from and to. Column numbers appear in inverse print along the top. As in the real game, you may not move the checkers backward.

This puzzle can be solved in 52 moves. Once again, you can press F1 for Auto-Solve.■

Please send me back issues of ReRUN

____ January/February 1986
____ March/April 1986
____ May/June 1986
____ July/August 1986
____ Productivity Pak II
____ September/October 1986
____ November/December 1986
____ January/February 1987
____ March/April 1987

____ Disk version(s) at $16.47 each*

* *Price include postage and handling. For foreign air mail, please add U.S. $1.50 per item Prepayment only.*

☐ Payment Enclosed ☐ MC ☐ VISA ☐ AE

Card # Exp. Date

Name

Address

City State Zip

Signature

ReRUN • 80 Elm Street • Peterborough, NH • 03458

BEAT THE RUSH!

Please send me:

☐ **1 year (6 issues) for $69.97**
☐ **July/August 1987 ReRUN disk for $16.47.***

Available in August 1987.

Includes programs for C-64 and C-128 (in both 64 and 128 modes).

Price includes postage and handling. For foreign air mail, please add U.S. $1.50 per item and $25 per subscription. Prepayment only. Subscriptions will start with the current issue.

☐ Payment Enclosed ☐ MC ☐ VISA ☐ AE

Card # Exp. Date

Name

Address

City State Zip

Signature

ReRUN • 80 Elm Street • Peterborough, NH • 03458

14 Programs Included on this Disk:

Windowing Utilities ► Notepad DOS ►
Mini Word Processor ► Program Debugger ►
Drum Machine ► Sprite Print Utility
Education ► Games

From the May RUN:

- 80-Column Custom Windows
- Disk Stuffer
- Sprite Print
- 128 Keys in 64 Mode
- PrintRite 64
- Handy Window

From the June RUN:

- Notepad DOS
- Drummodore
- Power Key
- Basic Bug Trap
- Robotgraph
- Presto Write 128
- The Shape of Sprites to Come

Bonus Program:

- Puzzler's Choice

If any manufacturing defect becomes apparent, the defective disk will be replaced free of charge if returned by prepaid mail within 30 days of purchase. Send it, with a letter specifying the defect, to:

ReRUN • 80 Elm Street • Peterborough, NH 03458

Replacements will not be made if the disk has been altered, repaired or misused through negligence, or if it shows signs of excessive wear or is damaged by equipment.

The programs in ReRUN are taken directly from listings prepared to accompany articles in *RUN* magazine. They will not run under all system configurations. Use the RUN It Right information included with each article as your guide.

The entire contents are copyrighted 1987 by CW Communications/Peterborough. Unauthorized duplication is a violation of applicable laws.

©Copyright 1987 CW Communications/Peterborough

CW COMMUNICATIONS/PETERBOROUGH